AMENDMENTS TO THE SPECIFICATION

Please amend paragraph [0016] as follows:

Figure 1 is ~~an exemplary executable~~source code within a program. As illustrated in Figure 1, in one embodiment, the ~~exemplary executable~~source code 100 includes multiple lines, each line containing a separate executable instruction. The executable instructions can be at a low backend level, or at some higher level closer to the front end. As shown in Figure 1, the ~~executable~~ source code 100 includes an "if(p)" clause and an "else" clause, each containing multiple executable instructions.

Please amend paragraph [0021] as follows:

Figure 3 is a flow diagram of one embodiment of the method to reduce the size of the executable code. As illustrated in **Figure 3**, at processing block 310, fork subgraph structures are identified within a graph structure, which represents the executable code. In one embodiment, the ~~executable~~ source code 100 is scanned for candidate fork subgraph structures. Each fork subgraph structure contains multiple tines or basic blocks, which share a common successor handle or block (in the case of the omega motion procedure) or which share a common predecessor handle or block (in the case of the alpha motion procedure).

Please amend paragraph [0027] as follows:

Referring back to **Figure 3**, at processing block 330, a data dependence graph structure is constructed for the ~~executable~~ source code. In one embodiment, for an alpha motion procedure, the data dependence graph structure has a directed arc u→v, if the instruction u must precede the instruction v, typically because the instruction v uses a value computed by the instruction u, i.e.

instruction v depends upon instruction u. Alternatively, for an omega motion procedure, the data dependence graph structure has a directed arc u→v, if the instruction u must succeed the instruction v. An arc u→v is said to be "properly oriented" within a tine t if instructions u and v belong to tine t, and if, when performing alpha motion, instruction u precedes v, or when performing omega motion, instruction u succeeds instruction v. (An improperly oriented arc can arise from loop-carried dependencies.)

Please amend paragraph [0028] as follows:

Figure 5 is a block diagram of a data dependence graph structure constructed in connection with the executable code. As illustrated in Figure 5, the data dependence graph structure 500 shows the directed edges between any two instructions within the ~~exemplary executable~~source code 100. For example, considering an alpha motion procedure, the foo(&x) instruction must precede the z=x*y instruction, the y=3 instruction must precede the z=x*y instruction, the a=3 instruction must precede the z=a*b instruction, and the foo(&b) instruction must precede the z=a*b instruction.